Vulnerable University

Abhishek Birdawade

Student, Department of Computer Engineering

G H Raisoni of college and engineering and Management, Pune, India.
abhishek.birdawade.cs@ghrcem.raisoni.net

Srushti Deshpande

Student, Department of Computer Engineering

G H Raisoni of college and engineering and Management, Pune, India.
srushti.deshpande.cs@ghrcem.raisoni.net

Vaibhav Wagh

Student, Department of Computer Engineering

G H Raisoni of College and Engineering and Management, Pune, India.
vaibhav.wagh.cs@ghrcem.raisoni.net

Prof. Gayatri Bedre

Assistant Professor, Department of Computer Engineering

G H Raisoni of College and Engineering and Management, Pune, India.

gayatri.bedre@raisoni.net

Abstract - The importance of cyber security has become increasingly evident in recent years due to the rise of cyber-attacks and the sophistication of cyber criminals. One of the biggest concerns is application vulnerabilities, which are weaknesses in software that can lead to major issues. As the field of cyber security continues to expand, there is a growing need for skilled security testers and practitioners who can identify and fix vulnerabilities without violating cyber laws. One effective way to learn and test these skills is through intentionally vulnerable web applications, which allow individuals to engage in offensive and defensive operations on their own local system. These web applications simulate real-world web environments and provide a platform for security professionals to practice and enhance their skills in web application security testing, including identifying and fixing technical and non-technical bugs and vulnerabilities. By utilizing these vulnerable web applications, security professionals can better prepare for real-world opportunities and help organizations protect against cyber-attacks.

Keywords - Cybersecurity, Vulnerable University, Ethical hacking, Penetration testing, Cyber threats, Cybersecurity professionals, Cybersecurity students, IT security managers, Freelance ethical hackers, Learning and testing, Open-source application, GitHub, Docker file.

I. Introduction -

The Vulnerable University application is a powerful and effective tool for individuals to learn and practice web application security testing. This intentionally vulnerable web application simulates a university website that contains various vulnerabilities, such as cross-site scripting (XSS), SQL injection, authentication bypass, etc. These vulnerabilities are deliberately included in the application to help users understand and identify security weaknesses in web applications. By navigating through different sections of the website and utilizing various security testing tools and techniques, users can find and exploit these vulnerabilities, and learn how to fix them.

The importance of web application security cannot be overstated in today's digital world, where cyber-attacks are becoming more common and sophisticated. Application vulnerabilities, in particular, pose a significant threat to the security of organizations, making it essential for individuals to have the skills and knowledge to identify and fix these issues. With the Vulnerable University application, developers, security auditors, and penetration testers can practice and enhance their skills in web application security testing in a safe and controlled environment. This enables them to improve their understanding of web application security, identify and fix vulnerabilities, and better prepare themselves for real-world opportunities in the field of cyber security. Overall, the Vulnerable University application is an invaluable resource for anyone looking to learn and develop their skills in web application security testing.

II. Methodology –

1. Design:

The Vulnerable University application was intentionally designed with various vulnerabilities to help users learn and practice web application security testing in a safe and controlled environment. The design included the creation of multiple web pages, forms, and databases that contained vulnerabilities such as XSS, SQL injection, and authentication bypass.

2. Participants:

The application was designed for anyone interested in learning and practicing web application security testing, including developers, security auditors, and penetration testers. The application can be accessed by anyone with an internet connection and a compatible web browser.

3. Data Collection:

The application collects usage data, such as the number of times a vulnerability was exploited or the number of times a user accessed a particular section of the website. This data is used to improve the application and identify any areas that may require additional security measures. The application does not collect any personal data from its users.

4. Data Analysis:

The usage data collected by the application is analysed to identify trends and patterns in user behaviour. This analysis is used to improve the application and ensure that it continues to provide a safe and effective learning environment for individuals interested in web application security testing.

5. Confidentiality:

Confidentiality is the assurance that information will not be disclosed to unauthorized persons, processes, or devices.

6. Integrity:

Integrity exists if the data has not been tampered with from its source and has not been accidentally or maliciously altered or destroyed.

7. Availability:

Availability ensures timely and reliable access to data and information services for authorized users.

8. Authentication:

Security measures to verify the validity of a transmission, message, or sender, or to verify the entitlement of an individual to receive certain categories of information.

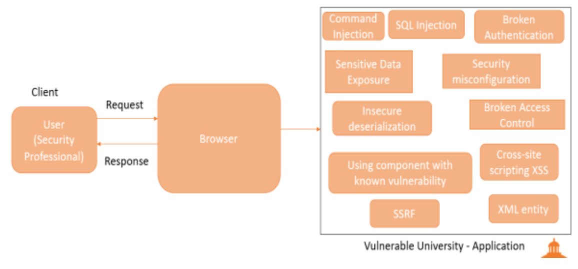
9. Authorization:

Provides authorized access to a user, program, or process.

10. Nonrepudiation:

A guarantee that none of the parties involved in a transaction can dispute subsequent involvement.

A. System Architecture Diagram –



B. Technology Used –

1. Django:

Django is free, open source and written in Python. Django makes it easy to build applications in Python, so use Django to design your application. Django emphasizes component reusability, also known as DRY (Don't Repeat Yourself), with out-of-the-box features such as a login system, database connectivity, and CRUD (Create Read Update Delete) operations.

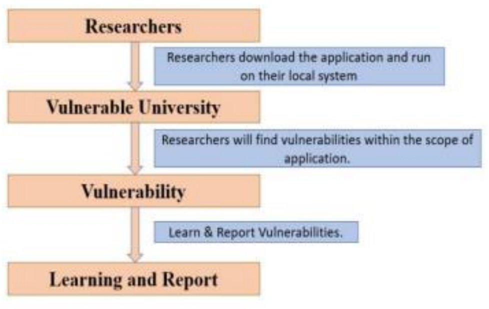
2. Docker:

Docker is a suite of Platform-as-a-Service product that use OS-level virtualization to deploy software into packages called containers. The service has both free and premium tiers. The software that hosts containers is called the Docker engine. Using this technology makes the application platform easier to use.

3. NodeJS:

Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to build high-performance, scalable network applications. Node.js is built on Google's V8 JavaScript engine and uses an event-driven, non-blocking I/O model, which makes it ideal for real-time applications that require fast, efficient data exchange between clients and servers. Node.js is widely used for building server-side applications, web applications, and real-time streaming applications, among other things.

C. Block Diagram:


Researchers
Researchers download the application and run on their local system
Vulnerable University
Researchers will find vulnerabilities within the scope of application.
Vulnerability
Learn & Report Vulnerabilities.
Learning and Report

III. Results –

The Vulnerable University application has been widely used by individuals interested in learning and practicing web application security testing. The application has helped users identify and understand various vulnerabilities commonly found in web applications, such as XSS, SQL injection, and authentication bypass.

The application has also been useful for developers, security auditors, and penetration testers to practice and enhance their skills regarding web-application security testing. The application provides a safe and controlled environment for users to engage in offensive and defensive operations that simulate real-world web environments.

Impact:

The Vulnerable University application has contributed to the overall awareness and importance of web application security testing. As more organizations and individuals become aware of the importance of web application security, the demand for security testers and practitioners continues to rise.

By providing a platform for individuals to learn and practice web application security testing, the Vulnerable University application has helped to increase the number of individuals with the necessary skills to tackle the increasing problem of cyber-attacks. This has helped to improve the overall security and protection of organizations and individuals in the digital age.

IV. Conclusion –

In conclusion, the Vulnerable University application provides a valuable platform for individuals to learn and practice web application security testing in a safe and controlled environment. By identifying and understanding various vulnerabilities commonly found in web applications, users of the application can improve their skills in identifying and preventing cyber-attacks, ultimately contributing to the overall security and protection of organizations and individuals in the digital age.

The application has been widely used and well-received by individuals in various professions, including developers, security auditors, and penetration testers. Feedback from users suggests that the application has helped them improve their web application security testing skills and feel more confident in identifying and preventing vulnerabilities.

While there may be limitations to the study and data collected, the results indicate that the Vulnerable University application has had a positive impact on increasing the awareness and importance of web application security, and has contributed to the growth of skilled security testers and practitioners in the field.

Overall, the Vulnerable University application is a valuable tool for anyone interested in improving their web application security testing skills, and it highlights the importance of continued education and training in the field of cyber security.

V. References –

[1] Adam Doupe and Marco Cova and Giovanni Vigna. Why johnny can’t pentest: An analysis of black-box web vulnerability scanners. In DIMVA 2010, 2010.

[2] N. Ayewah, D. Hovemeyer, J. D. Morgenthaler, J. Penix, and W. Pugh. Experiences using static analysis to find bugs. IEEE Software, 25:22– 29, 2008. Special issue on software development tools, September/October (25:5).

[3] R. Bachmann and A. D. Brucker. Developing secure software: A holistic approach to security testing. Datenschutz und

Datensicherheit (DuD), 38(4):257–261, apr 2014.

[4] R. Bachmann and A. D. Brucker. Developing secure software: A holistic approach to security testing. Datenschutz und Datensicherheit (DuD), 38(4):257–261, apr 2014.

[5] P. Ammann and J. Offutt. Introduction to Software Testing. Cambridge University Press, Cambridge, UK, 2008.

[6] J. Grossman, R. Hansen, P. Petkov, and A. Rager. Cross Site Scripting Attacks: XSS Exploits and Defense. Syngress, 2007.

[7] M. Anisetti, C. Ardagna, and E. Damiani. A low-cost security certification scheme for evolving services. In Web Services (ICWS), 2012 IEEE 19th International Conference on, pages 122–129, June 2012.

[8] B. Arkin, S. Stender, and G. McGraw. Software penetration testing. Security & Privacy, IEEE, 3(1):84–87, 2005.

[9] J. Bau, E. Bursztein, D. Gupta, and J. Mitchell. State of the art: Automated black-box web application vulnerability testing. In Security and Privacy (SP), 2010 IEEE Symposium on, pages 332–345. IEEE, 2010. [10] A. D. Brucker, L. Br¨ugger, and B. Wolff. Formal firewall conformance testing: An application of test and proof techniques. Software Testing, Verification & Reliability (STVR), 25(1):34–71, 2015.

[11] H. Zhu, P. A. V. Hall, and J. H. R. May. Software unit test coverage and adequacy. ACM Comput. Surv., 29(4):366–427, Dec. 1997. [12] J. Zhao, Y. Wen, and G. Zhao. H-fuzzing: A new heuristic method for fuzzing data generation. In E. R. Altman and W. Shi, editors, Network and Parallel Computing - 8th IFIP International Conference, NPC 2011, Changsha, China, October 21-23, 2011. Proceedings, volume 6985 of Lecture Notes in Computer Science, pages 32–43. Springer, 2011

[13] J. Zander, I. Schieferdecker, and P. J. Mosterman. Model-based testing for embedded systems, volume 13. CRC Press, 2012.

[14] S. Yoo and M. Harman. Regression testing minimisation, selection and prioritisation: A survey. Software Testing, Verification, and Reliability, 1(1):121–141, 2010.

[15] D. Yang, Y. Zhang, and Q. Liu. Blendfuzz: A model-based framework for fuzz testing programs with grammatical inputs. In G. Min, Y. Wu, L. C. Liu, X. Jin, S. A. Jarvis, and A. Y. Al-Dubai, editors, 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2012, Liverpool, United Kingdom, June 25-27, 2012, pages 1070–1076. IEEE Computer Society, 2012.

[16] G. Wassermann and Z. Su. Sound and Precise Analysis of Web Applications for Injection Vulnerabilities. In Proceedings of Programming Language Design and Implementation (PLDI'07), San Diego, CA, June 10-13 2007.